

# Specyfikacja wymagań systemowych

3@KASK

25 maja 2009

Symbol projektu: 3@KASK	Opiekun projektu: mgr inż. Tomasz Boiński
Nazwa Projektu: Wizualizacja grafów za pomocą biblioteki Prefuse	

Nazwa Dokumentu: Specyfikacja wymagań systemowych	Nr wersji: 0.4
Odpowiedzialny za dokument: Piotr Orłowski	Data pierwszego sporządzenia: 15 kwietnia 2009
Przeznaczenie: DLA KLIENTA	Data ostatniej aktualizacji: 25 maja 2009

## Historia dokumentu

Wersja	Opis modyfikacji	Rozdział/strona	Autor modyfikacji	Data
1	Stworzenie	wszystkie	Grupa projektowa	15.04.09
2	Wpisanie celów i wymogów ogólnych	cele	Grupa projektowa	16.04.09
3	Wpisanie funkcjonalności wizualizacyjnych		Grupa projektowa	28.04.09
4	Opis wymagań		Grupa projektowa	05.05.09
5	Zmiana kolorów Property (SomeValuesFrom i AllValuesFrom)	Projekt wizualizacji	Grupa projektowa	18.05.09
6	WJ001 - klasa Thing w grafie	Wymagania jakościowe	Grupa projektowa	25.05.09

## Spis treści

<b>1</b>	<b>Cele systemu</b>	<b>3</b>
1.1	Cele biznesowe . . . . .	3
1.2	Cele funkcjonalne . . . . .	4
<b>2</b>	<b>Otoczenie systemu</b>	<b>4</b>
2.1	Użytkownicy . . . . .	4
2.2	Systemy zewnętrzne . . . . .	4
<b>3</b>	<b>Przewidywane komponenty systemu</b>	<b>4</b>
3.1	Podsystemy . . . . .	4
3.2	Komponenty sprzętowe . . . . .	4
3.3	Programowe . . . . .	5
<b>4</b>	<b>Wymagania funkcjonalne</b>	<b>5</b>
4.1	Wymagania wizualizacji ontologii . . . . .	6
4.2	Projekt wizualizacji . . . . .	7
<b>5</b>	<b>Wymagania na dane</b>	<b>8</b>
<b>6</b>	<b>Wymagania jakościowe</b>	<b>9</b>
6.1	Wymagania w zakresie wiarygodności . . . . .	9
6.2	Wymagania w zakresie wydajności . . . . .	9
6.3	Wymagania w zakresie elastyczności . . . . .	9
6.4	Wymagania w zakresie użyteczności . . . . .	9
<b>7</b>	<b>Sytuacje wyjątkowe</b>	<b>9</b>
<b>8</b>	<b>Dodatkowe wymagania</b>	<b>9</b>
8.1	Wymagania sprzętowe . . . . .	9
8.2	Wymagania programowe . . . . .	10
8.3	Inne wymagania . . . . .	10
<b>9</b>	<b>Kryteria akceptacyjne</b>	<b>10</b>
	<b>Literatura</b>	<b>11</b>

# 1 Cele systemu

## 1.1 Cele biznesowe

CB001	Ułatwienie pracy programistom tworzącym aplikacje wizualizujące ontologie
Opis:	Istnieje zapotrzebowanie na bibliotekę tłumaczącą OWL bezpośrednio na elementy graficzne.
Źródło:	Wstępna specyfikacja projektu
Priorytet:	bardzo ważne

CB002	Ułatwienie zakończenia projektu OCS
Opis:	Moduł wizualizujący ontologie w OCS wymaga modernizacji i rozbudowy funkcjonalności. Zapewnienie biblioteki wizualizującej ontologie ułatwi i przyspieszy ten proces.
Źródło:	Klient - mgr inż. Tomasz Boiński
Priorytet:	bardzo ważne

CB003	Zwiększenie atrakcyjności portalu OCS
Opis:	Poprawa estetyki modułu wizualizującego ontologię może przyczynić się do sukcesu portalu po jego wdrożeniu.
Źródło:	Klient - mgr inż. Tomasz Boiński
Priorytet:	mało ważne

## 1.2 Cele funkcjonalne

CF001	Intuicyjne API
Opis:	Na podstawie subiektywnej opinii kilku zewnętrznych osób nie należących do zespołu projektowego oraz oceny opieku- na projektu API powinno zostać uznane za nadające się do użytku.
Źródło:	Klient - mgr inż. Tomasz Boiński
Priorytet:	średnio ważne

CF002	Dobra dokumentacja
Opis:	Przygotowanie dokumentacji w Javadoc ułatwi pracę użyt- kownikom biblioteki.
Źródło:	Klient - mgr inż. Tomasz Boiński
Priorytet:	bardzo ważne

CF003	Wizualizacja ontologii
Opis:	Stworzenie biblioteki, która pozwoli na wizualizację obiek- tów OWL API przy użyciu odpowiedniej biblioteki graficz- nej.
Źródło:	Specyfikacja projektu
Priorytet:	bardzo ważne

CF004	Umożliwienie graficznej edycji i dodawania obiektów OWL API
Opis:	Dostarczenie tej funkcjonalności ułatwi tworzenie progra- mów z interfejsem pozwalającym na edycję ontologii zapi- sanych w OWL API.
Źródło:	Klient - mgr inż. Tomasz Boiński
Priorytet:	średnio ważne

CF005	Udostępnienie informacji do debuggowania
Opis:	Biblioteka powinna wysyłać komunikaty informacyjne, ostrzegawcze oraz informujące o błędach na strumień udo- stępniiony użytkownikowi.
Źródło:	Standard tworzenia biblioteki [2]
Priorytet:	średnio ważne

## 2 Otoczenie systemu

### 2.1 Użytkownicy

Specyfika projektu nie definiuje użytkowników systemu.

### 2.2 Systemy zewnętrzne

Specyfika systemu nie wymaga definiowania systemów zewnętrznych.

## 3 Przewidywane komponenty systemu

### 3.1 Podsystemy

Specyfika projektu sprawia, że podsystemy nie będą rozpatrywane.

### 3.2 Komponenty sprzętowe

Specyfika projektu sprawia, że komponenty sprzętowe nie będą rozpatrywane.

### 3.3 Programowe

KS001	Prefuse
Opis:	Biblioteka graficzna do wizualizacji grafów w języku Java
Powiązania:	
Źródło:	Specyfikacja projektu
Priorytet:	bardzo ważne

## 4 Wymagania funkcjonalne

WF001	Udostępnienie kilku algorytmów wizualizacji
Opis:	Biblioteka powinna udostępniać kilka trybów prezentacji grafów (np. w formie drzewa, w formie gwiazdy i innych).
Dotyczy:	CF003
Źródło:	klient - mgr Tomasz Boiński
Powiązania:	WF002
Priorytet:	ważny

WF002	Parametryzacja trybów wizualizacyjnych
Opis:	Domyślne parametry w trybach wizualizacji (takie jak długość krawędzi grafu, automatyczne układanie) powinny zostać dobrane w taki sposób, by obraz był przejrzysty, stabilny i czytelny.
Dotyczy:	CF003
Źródło:	klient - mgr Tomasz Boiński
Powiązania:	WF001
Priorytet:	średnio ważny

WF003	Udostępnienie strumienia błędów
Opis:	Biblioteka będzie udostępniać strumień danych, w którym znajdują się komunikaty o błędach. Strumień ten będzie mógł zostać wykorzystany przez użytkownika.
Dotyczy:	CF005
Źródło:	klient - mgr inż. Tomasz Boiński
Powiązania:	
Priorytet:	ważne

WF010	Dodatkowe informacje
Opis:	Biblioteka będzie dostarczać informacje o wersji ontologii zapisane w pliku OWL oraz dodatkowe informacje o klasach (annotationProperty).
Dotyczy:	CF003
Źródło:	klient - mgr inż. Tomasz Boiński
Powiązania:	
Priorytet:	średnio ważne

## 4.1 Wymagania wizualizacji ontologii

WF004	Rozróżnialność podstawowych symboli
Opis:	Class, Individual, Property powinny mieć rozróżnialne symbole
Dotyczy:	CF003
Źródło:	klient - mgr inż. Tomasz Boiński
Powiązania:	
Priorytet:	bardzo ważne

WF005	Rozróżnialność szczególnych typów Class
Opis:	Klasa anonimowa, datatype, Thing i Nothing powinny być łatwo rozpoznawalne.
Dotyczy:	CF003
Źródło:	klient - mgr inż. Tomasz Boiński
Powiązania:	WF004
Priorytet:	ważne

WF006	Rozróżnialność związków między klasami (Class), instancjami (Individual) oraz predykatami (Property)
Opis:	Różne symbole dla equivalentClass, disjointWith, subclassOf, sameAs, differentFrom, allDifferent, oneOf, unionOf, intersectionOf, complementOf, subProperty, equivalentProperty, hasProperty.
Dotyczy:	CF003
Źródło:	klient - mgr inż. Tomasz Boiński
Powiązania:	WF005, WF004
Priorytet:	ważne

WF007	Rozróżnialność ograniczeń predykatów (Restrictions)
Opis:	Wyróżnić kardynalność (cardinality), domeny (domains) predykatów, inverseOf, właściwości predykatów (transitive, symmetric, functional, inverseFunctional).
Dotyczy:	CF003
Źródło:	klient - mgr inż. Tomasz Boiński
Powiązania:	WF004
Priorytet:	ważne

WF008	Podświetlanie wybranych związków i powiązań.
Opis:	Podświetlać subklasy danej klasy po ich wybraniu myszką po zdefiniowanym zdarzeniu; podobnie subproperty i complex class.
Dotyczy:	CF003
Źródło:	klient - mgr inż. Tomasz Boiński
Powiązania:	WF006
Priorytet:	mało ważne

WF009	Możliwość definiowania zdarzeń.
Opis:	Użytkownik będzie mógł pod uchwyty zdarzeń podpinąć własne funkcje obsługi.
Dotyczy:	CF003, CF004
Źródło:	klient - mgr inż. Tomasz Boiński
Powiązania:	
Priorytet:	mało ważne

## 4.2 Projekt wizualizacji

Identyfikator:	Nazwa	Wizualizacja
PW001:	Thing	
PW002:	Nothing	
PW003:	Class	
PW004:	Individual	
PW005:	Property	
PW006:	Datatype	
PW007:	Anonymous Class	
PW008:	Subclass	
PW009:	instanceOf	
PW010:	equivalentClass	
PW011:	disjointWith	
PW012:	differentFrom / allDifferent	
PW013:	sameAs	
PW014:	oneOf	
PW015:	unionOf	
PW016:	intersectionOf	
PW017:	complementOf	

PW018:	subProperty	
PW019:	inverseOf (property)	
PW020:	equivalentProperty	
PW021:	functionalProperty	
PW022:	inverseFunctionalProperty	
PW023:	symmetricProperty	
PW024:	transitiveProperty	
PW025:	hasProperty	
PW026:	domain	
PW027:	range	
PW028:	allValuesFrom	
PW029:	someValuesFrom	
PW030:	minCardinality / maxCardinality	
PW031:	cardinality	

## 5 Wymagania na dane

WD001	Obsługa obiektów OWL API
Opis:	Biblioteka będzie przystosowana do pobierania, obróbki i zwracania obiektów OWL API.
Powiązania:	
Źródło:	Klient - mgr inż. Tomasz Boiński
Priorytet:	bardzo ważne



## 6 Wymagania jakościowe

### 6.1 Wymagania w zakresie wiarygodności

WJ001	Poprawność wizualizacji
Opis:	Wszystkie wizualizowane elementy powinny pochodzić z ontologii otrzymanej na wejściu programu. Program nie powinien dodawać własnych elementów (np. wywnioskowanych). Wyjątkowo dla klas, które nie mają zdefiniowany nadklas zostanie utworzony związek z klasą Thing.
Powiązania:	WJ002
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	bardzo ważne

WJ002	Kompletność wizualizacji
Opis:	Jeżeli biblioteka nie wizualizuje danej funkcji OWL API informacja o tym powinna znaleźć się w strumieniu błędów.
Powiązania:	CF005, WJ001, WD001
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	ważne

### 6.2 Wymagania w zakresie wydajności

Brak wymogów wydajnościowych ze względu na specyfikę projektu.

### 6.3 Wymagania w zakresie elastyczności

WJ003	Obsługiwane wersje Javy
Opis:	Biblioteka powinna wspierać wersje Javy 1.5 i nowsze.
Powiązania:	
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	bardzo ważne

WJ004	Obsługiwane wersje OWL API
Opis:	Powinna istnieć możliwość podpięcia zewnętrznego OWL API (wybranego przez użytkownika/programistę).
Powiązania:	
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	bardzo ważne

### 6.4 Wymagania w zakresie użyteczności

Ze względu na przyjętą metodykę wytwarzania oprogramowania zagadnienie to zostanie rozpatrzone w przyszłości.

## 7 Sytuacje wyjątkowe

Ze względu na specyfikę projektu sytuacje wyjątkowe nie będą rozpatrywane.

## 8 Dodatkowe wymagania

### 8.1 Wymagania sprzętowe

Ze względu na specyfikę projektu wymagania sprzętowe nie będą rozpatrywane.

## 8.2 Wymagania programowe

WD003	JVM
Opis:	Do skorzystania z biblioteki niezbędna jest JVM.
Dotyczy:	CF001, CF002
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	ważne

## 8.3 Inne wymagania

WD001	Dokumentacja w javadoc
Opis:	Wszystkie ważne klasy i funkcje powinny mieć odpowiednią dokumentację w formacie javadoc.
Dotyczy:	CF001, CF002
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	ważne

WI002	Dokumentacja w języku angielskim
Opis:	Dokumentacja wszystkich funkcji i klas powinna posiadać angielską wersję językową.
Dotyczy:	CF001, CF002
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	mało ważne

WI003	Dokumentacja w języku polskim
Opis:	Dokumentacja wszystkich funkcji i klas powinna posiadać polską wersję językową.
Dotyczy:	CF001, CF002
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	ważne

WI004	Nazwy zmiennych i funkcji w języku angielskim
Opis:	Nazwy zmiennych i funkcji powinny zostać dobrane w języku angielskim i zgodnie ze standardami programowania w javie [1].
Dotyczy:	CF001, CF002
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	ważne

## 9 Kryteria akceptacyjne

KA001	Spełnione są podstawowe wymagania wymienione w dokumencie SWS
Opis:	Spełnione są wszystkie wymagania ważne i bardzo ważne zdefiniowane w SWS.
Dotyczy:	W*
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	ważne

KA002	Biblioteka współpracuje z OWL API dostarczonym przez KASK
Opis:	Biblioteka współpracuje z OWL API dostarczonym przez KASK zbudowanym na podstawie OWL API ver 2.1.1
Dotyczy:	WJ004
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	ważne

## Literatura

- [1] Code conventions for the javatm programming language. publikacja elektroniczna, kwiecień 1999.  
<http://java.sun.com/docs/codeconv/html/CodeConventions.doc8.html>.
- [2] Greg Travis. Build your own java library. publikacja elektroniczna.  
[http://www.digilife.be/quickreferences/PT/Build your own Java library.pdf](http://www.digilife.be/quickreferences/PT/Build%20your%20own%20Java%20library.pdf).